Precise Image-Based Motion Estimation for Autonomous Small Body Exploration

Andrew E. Johnson and Larry H. Matthies
Autonomy and Control Section
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA 91109

Abstract

Space science and solar system exploration are driving NASA to develop an array of small body missions ranging in scope from near body flybys to complete sample return. This paper presents an algorithm for onboard motion estimation that will enable the precision guidance necessary for autonomous small body landing. Our techniques are based on automatic feature tracking between a pair of descent camera images followed by two frame motion estimation and scale recovery using laser altimetry data. The output of our algorithm is an estimate of rigid motion (attitude and position) and motion covariance between frames. This motion estimate can be passed directly to the spacecraft guidance and control system to enable rapid execution of safe and precise trajectories.

1 Introduction

Due to the small size, irregular shape and variable surface properties of small bodies, accurate motion estimation is needed for safe and precise small body exploration. Because of the communication delay induced by the large distances between the earth and targeted small bodies, landing on small bodies must be done autonomously using on-board sensors and algorithms. Current navigation technology does not provide the precision necessary to accurately land on a small bodies, so novel motion estimation techniques must be developed. Computer vision offers a possible solution to precise motion estimation.

Historically, optical navigation has been used for orbit determination and instrument pointing during close fly-bys of small bodies and moons of the outer planets. Generally, this has been implemented by ground-based image processing to extract centroids of small reference targets like asteroids and moons from which target relative spacecraft attitude and position are computed.

The Near Earth Asteroid Rendezvous (NEAR), a current mission that will rendezvous with asteroid Eros 433 in February 2000, uses optical navigation extensively for orbit determination and small body 3-D modeling [5]. The baselined navigation technique will combine manually designated landmarks from imagery of Eros and radiometric data to compute and control the trajectory of the orbiter. The NEAR mission will clearly demonstrate the effectiveness of optical navigation. However, this ground-based paradigm will not map to missions involving small body exploration and landing.

Small body exploration requires multiple precise target relative maneuvers during a brief descent to the surface. The round trip light time prohibits the determination of the necessary trajectory control maneuvers on the ground. Furthermore, typical onboard position sensors do not have the accuracy needed for small body landing (e.g., during a small body descent taking a few hours accelerometer errors will grow to the kilometer level). However, the required positional accuracies can be obtained if autonomous real-time optical navigation methods are developed.

The Deep Space 1 mission as part of the New Millennium Program is flying an autonomous optical navigation technology demonstration. The DS-1 AutoOpNav system will use onboard centroiding of reference asteroids for autonomous navigation during small body fly-bys [6]. They expect to obtain automatic position estimates with accuracies on order of 100 kilometers. For scientific instrument pointing purposes, this accuracy is sufficient. Controlled small body landing will require much better position and motion estimation accuracies. Furthermore, since the appearance of the small body is variable, small body landing cannot always rely on reference landmarks for navigation. The DS-1 AutoOpNav system will demonstrate autonomy and computer vision in space, however for small body landing a more versatile and accurate system is required.

This paper describes a fully autonomous and onboard solution for accurate and robust motion estimation near a proximal small body. Our techniques are based on automatic feature tracking between a pair of images followed by two frame motion estimation and scale recovery using laser altimetry data. The output of our algorithm is an estimate of rigid motion (attitude and position) and motion covariance between frames. This motion estimate can be passed directly to the spacecraft guidance navigation and control system to enable rapid execution of safe and precise trajectories.

The work described in this paper was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under contract from the National Aeronautics and Space Administration.

2 Motion Estimation

Motion estimation from images has a long history in the machine vision literature. The algorithm presented in this paper falls in the category of two-frame feature-based motion estimation algorithms Once the spacecraft sensors are pointed at the small body surface, our algorithm works as follows. At one time instant a descent camera image and a laser altimeter reading are taken. A short time later, another image and altimeter reading are taken. Our algorithm then processes these pairs of measurements to estimate the rigid motion between readings. There are multiple steps in our algorithm. First, distinct features, which are pixels that can be tracked well across multiple images, are detected in the first image. Next, these features are located in the second image by feature tracking. Given these feature matches, the motion state and covariance of the spacecraft, up to a scale on translation, are computed using a two stage motion estimation algorithm. Finally the scale of translation is computed by combining altimetry with the motion estimates using one of two methods which depend on the descent angle. The block diagram for motion estimation is shown in Figure 1.

2.1 Feature Detection

The first step in two-frame motion estimation is the extraction of features from the first image. Features are pixel locations and the surrounding image intensity neighborhood (call this a feature window) that can be tracked well across multiple images that may under go arbitrary, but small, changes in illumination or viewing direction. A qualitative definition of a good feature is a feature window that has strong texture variations in all directions.

Feature detection has been studied extensively and multiple proven feature detection methods exist. Consequently, we elected to implement a proven feature detection method instead of redesigning our own. Since processing speed is a very important design constraint for our application, we selected the state of the art feature detection algorithm of Benedetti and Perona [2]. This algorithm is an implementation of the well know Shi-Tomasi feature detector and tracker [7] modified to eliminate transcendental arithmetic.

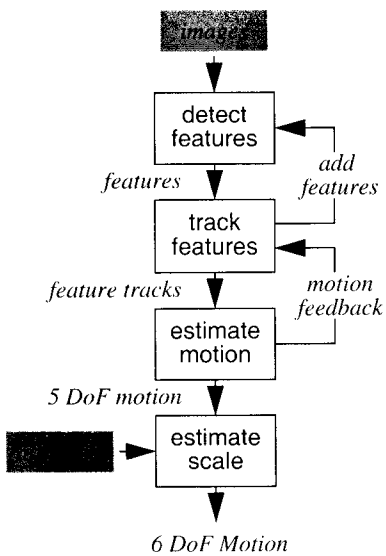

Figure 1: Block diagram for motion estimation.

Surfaces of small bodies generally appear highly textured, so good features to track are expected to be plentiful. Usually feature detection algorithms exhaustively search the image for every distinct feature. However, when the goal is motion estimation, only a relatively small number of features need to be tracked (~100). The speed of feature tracking can be increased up to two orders of magnitude by using a random search strategy, instead of an exhaustive search for all good features, while still guaranteeing that the required number of features are detected. Suppose that N features are needed for motion estimation. Our detection algorithm selects a pixel at random from the image. If the randomly selected pixel has an interest value greater than a predetermined threshold, it is selected as a feature. This procedure is repeated until N features are detected.

2.2 Feature Tracking

The next step in motion estimation is to locate the features detected in the first frame in the second frame. This procedure is called feature tracking. As with feature detection, there exist multiple methods for feature tracking in the machine vision literature. Feature tracking can be split in to two groups of algorithms: correlation based methods and optical flow based methods [7]. Correlation based methods are appropriate when the motion of features in the image is expected to be large. For small motions, optical flow based methods are more appropriate because in general they require less computation than correlation methods. We use the Shi-Tomasi feature tracker an optical flow based method for feature tracking, because in our application of precision landing, we know a-priori that the motion between image frames will be small. Our

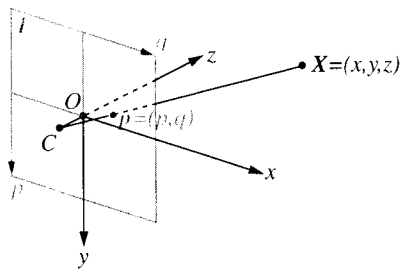

Figure 2: Unit focal length imaging geometry. World coordinate origin O is on image plane and optical center C is 1 unit behind image plane.

implementation of feature tracking follows that in [7] for 2-D (not affine) feature motion.

2.3 Two Frame Motion Estimation

The motion between two camera views can be described by a rigid transformation (R,T) where R encodes the rotation between views and T encodes the translation between views. Once features are tracked between images, the motion of the camera can be estimated by solving for the motion parameters that, when applied to the features in the first image, bring them close to the corresponding features in the second image.

In our algorithm, motion estimation is a two stage process. First an initial estimate of the motion is computed using a linear algorithm. This algorithm is applied multiple times using different sets of features to eliminate feature track outliers and determine a robust LMedS estimate of motion. The result of this algorithm is then used as input to a more accurate nonlinear algorithm that solves for the motion parameters directly. Since an good initial estimate is needed to initialize any nonlinear feature-based motion estimation algorithm, this two stage approach is common [11]. Output from the nonlinear algorithm is the estimate of the five motion parameters and their covariance. Our algorithm assumes that the camera taking the images has been intrinsically calibrated (i.e., focal length, radial distortion, optical center, skew and aspect are all known).

A fundamental short coming of all image-based motion estimation algorithms is the inability to solve for the magnitude of translational motion. Intuitively the reason for this is that the algorithms cannot differentiate between a very large object that is far from the camera or a small object that is close to the camera; the camera does not convey information about scene scale. Consequently, the output of motion estimation is a 5 DoF motion composed of the a unit vector $T_s = T/\|T\|$ describing the direction of heading and the rotation matrix R between views. As is shown in the next section, laser altimetry can be combined with 5 DoF motion estimation to compute the complete 6 DoF motion of the camera.

2.3.1. Robust Linear Motion Estimation

The first stage of motion estimation uses a linear algorithm to compute the motion between views [4]. Since the linear algorithm has a closed form solution, motion can be computed quickly. However, the linear algorithm does not solve for the motion parameters directly, so its results will not be as accurate as those obtained using the nonlinear algorithm. Our linear algorithm is an implementation of the algorithm presented in [10] augmented by normalization presented in [3] for better numerical conditioning. To filter out possible outliers in feature detection, we use a robust linear motion estimation algorithm based on least median of squares[12].

2.3.2. Nonlinear Motion Estimation

Robust linear motion estimation serves two purposes: it provides an initial estimate of the 5 DoF motion between views and it detects and eliminates feature track outliers. The nonlinear algorithm takes the initial linear estimate of the motion and refines it by minimizing an error term that is a function of the motion parameters and the outlier-free feature tracks. There exists many nonlinear motion estimation algorithms in the vision literature. Instead of starting from scratch, the nonlinear algorithm we have developed combines the attractive elements of multiple algorithms to produce an algorithm that is computationally efficient, numerically stable and accurate. For numerical stability, we use the camera model parameterization of Azarbayejani and Pentland[1]. For highly accurate motion parameter estimation we use the Levenberg-Marquardt algorithm as proposed by Szeliski and Kang[8]. Finally, for computational efficiency, we remove the scene structure from the nonlinear minimization as suggested by Weng et al. in [11].

First, the homogenous coordinates of each feature are determined by projecting them onto the unit focal plane. This projection will depend on the lens, imager, and camera model used. A simple model for the transformation of a feature at pixel location (p_i,q_i) to its homogenous coordinates u_i is

$$u_i = \begin{bmatrix} u_i & v_i & 1 \end{bmatrix}^T = \begin{bmatrix} \frac{p_i - C_p}{f} & \frac{q_i - C_q}{sf} & 1 \end{bmatrix}^T \quad (1)$$

where (C_p,C_q) is the center of the camera in pixel units, f is the focal length of the camera in pixel units and s is the aspect ratio of the pixels. This model assumes no radial distortion in the camera. More sophisticated models that include radial distortion are used when necessary [9].

Before we can express the error function, we need to detail the motion parameters over which the minimization will take place. First of all, the motion between frames is presented as a translation and rotation pair (R,T). To simplify the parameter estimation, we represent the rotation with a unit quaternion $q = \begin{bmatrix} q_0 & q_1 & q_2 & q_3 \end{bmatrix}^T$ where the rotation matrix in terms of a unit quaternion is

$$R(q) = \begin{bmatrix} q_o^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1q_2 - q_0q_3) & 2(q_1q_3 + q_0q_2) \\ 2(q_1q_2 + q_0q_3) & q_o^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2q_3 - q_0q_1) \\ 2(q_1q_3 - q_0q_2) & 2(q_2q_3 + q_0q_1) & q_o^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix} \quad (2)$$

The translation is represented by a unit vector $T = \begin{bmatrix} T_x & T_y & T_z \end{bmatrix}^T$. Together the unit quaternion and unit

translation comprise the parameter state vector $\boldsymbol{a}$.

$$\boldsymbol{a} = \begin{bmatrix} q_o & q_1 & q_2 & q_3 & T_x & T_y & T_z \end{bmatrix}^T \quad (3)$$

Nonlinear motion estimation attempts to minimize the image plane error between the features in the second view and the projection of the features in the first view into the second view given the motion between frames.

If the unit focal coordinates (defined by Equation 1) of the features in image I are $\boldsymbol{u}_i = \begin{bmatrix} u_i & v_i \end{bmatrix}^T$ and $\boldsymbol{u}_i' = \begin{bmatrix} u_i' & v_i' \end{bmatrix}^T$ in image J, then the image plane error is

$$C(a) = \sum_i \left\| \boldsymbol{u}_i' - f(\boldsymbol{u}_i, \boldsymbol{a}) \right\|^2 \quad (4)$$

where f represents the projection of the features $\boldsymbol{u}_i'$ into image J given the motion $\boldsymbol{a}$. Correct image projection requires knowledge of the depth to a feature and a perspective camera model. Using the model of Azarbayejani and Pentland [1], if the (unknown) feature depths from the image plane are α_i, then the relation between unit focal feature coordinates and 3-D feature coordinates is

$$\boldsymbol{X}_i = \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} = \begin{bmatrix} u_i(1+\alpha_i) \\ v_i(1+\alpha_i) \\ \alpha_i \end{bmatrix} \quad (5)$$

The features in image I are transformed into image J according to

$$X'_i = \begin{bmatrix} x'_i & y'_i & z'_i \end{bmatrix}^T = R(q)\boldsymbol{X}_i + \boldsymbol{T}. \quad (6)$$

By combining Equation 5 and Equation 6, the feature depths $\begin{bmatrix} \alpha_i & \alpha_i' \end{bmatrix}^T$ can be computed through triangulation by solving

$$\begin{bmatrix} -R\boldsymbol{u}_i & \boldsymbol{u}_i' \end{bmatrix} \begin{bmatrix} \alpha_i \\ \alpha_i' \end{bmatrix} = \boldsymbol{T} \quad (7)$$

assuming that the translation between views is nonzero[10].

The camera model given the imaging geometry, shown in Figure 2, is

$$f(\boldsymbol{u}'_i, \boldsymbol{a}) = \begin{bmatrix} x' \\ y' \end{bmatrix} \frac{1}{1+z'} \quad (8)$$

Combining Equation 5, Equation 6, and Equation 8 results in a complete definition of Equation 4.

To estimate the motion parameters, we minimize Equation 4 using the Levenberg-Marquardt algorithm for nonlinear minimization. This approach was also used by Szeliski and Kang [8], however, unlike in their approach, we do not include the feature depths in the minimization. Inclusion of the feature depths would increase the length of the parameter vector from 7 to 7+N. Since the minimization relies on an inversion of a square matrix of rank equal to the length of the parameter vector, a computationally expensive matrix inversion would result. Since feature depths can be computed directly from the motion between views, it is not necessary to include them in the parameter vector. Instead, at each iteration, the feature depths are updated using the current motion estimate.The result is a computationally efficient and accurate motion estimation algorithm.

Since we are solving for a rotation represented by a unit quaternion and also a unit length translation, these constraints need to be enforced during minimization. We enforce these constraints by setting $\|q + \delta q\| = 1$ and $\|T + \delta T\| = 1$ during the update of the parameter vector at each iteration of the Levenberg-Marquardt algorithm. Consequently, these constraints are enforced while not complicating the minimization by including the constraints explicitly in the minimization function.

The output of nonlinear motion estimation is an estimate of the 5 DoF motion between views. In addition, the covariance Σ of the motion parameters $\boldsymbol{a}$ can be extracted directly from the quantities computed during minimization using

$$\Sigma(a) = A^{-1}. \quad (9)$$

2.4 Scale Computation Using Altimeter

The final stage of motion estimation computes the remaining motion parameter, magnitude of translation, from laser altimetry data. Depending on descent angle and surface relief, one of two complimentary methods is used.

2.4.1. Motivation

Motion estimation using monocular imagery cannot solve directly for the magnitude of translation, so an external means must be used to recover this parameter. For a spacecraft in orbit about a small body, there exist multiple possible solutions.

One solution is to integrate the accelerometer measurement in the spacecraft inertial reference unit to determine position. The advantage of accelerometers is that they present a completely onboard solution. Unfortunately, because that come from integration of noisy acceleration measurements, position measurements from accelerometers may be too inaccurate for precision landing.

The traditional approach is to use radiometric tracking measurements from earth. This approach has the advantage that it is well understood and uses equipment already on board the spacecraft. However, radiometric tracking has many disadvantages. First, it requires dedicated Deep Space Network tracking which is expensive and difficult to schedule. Second, round trip light time for tracking from earth induces a large latency in any position measurements (approximately 24 minutes for comet Tempel 1).

Multiple missions have or are using laser altimeters for

science return and navigation. As shown below, laser altimeters can also be used as a navigation sensor by aiding the determination of the position of the spacecraft. Laser altimeters give accurate range estimates and, when combined with a descent imager, present a complete on-board solution to 6-D body relative motion estimation. A disadvantage of the laser altimeter approach is that they have limited range (50 km for the NEAR laser altimeter). However, near body operations is precisely when accurate position estimation is needed the most, so this is not a major issue. A laser altimeter is an additional sensor; however, science return combined with navigational use justify the addition. Based on the disadvantages of the other available options, we determined that the use of a laser altimeter was the most promising solution for scale estimation.

2.4.2. Difference Scale Estimation

If images are taken as the spacecraft descends vertically to the surface, or the surface has very little surface relief, computation of translation magnitude is straightforward. Laser altimeter readings A_I and A_J are acquired simultaneously with each image. As shown in Figure 3, the difference in altimeter readings is equal to the translation of the spacecraft along the z-axis between images. Consequently, the magnitude of translation is

$$\|T\| = \frac{(A_I - A_J)}{t_z} \tag{10}$$

For motion approaching horizontal, t_z approaches zero, Equation 10 becomes ill conditioned and difference scale estimation will not work. Furthermore, if the spacecraft is not descending vertically and the surface topography is rough on order of the scale of translation then the difference of altimeter readings will not accurately reflect the z component of the translation. Once again, difference scale estimation will not work. Fortunately a different, albeit more complicated, procedure exists for computing scale in these cases.

2.4.3. Structure-Based Scale Estimation

From the feature-based motion estimate, the scaled depths α_i (Equation 7) to features in the scene can be computed. Assuming, without loss of generality, that the laser altimeter is aligned with the camera optical axis, features in the optical center will be at a depth equivalent to the laser altimeter reading. Consequently, the ratio of the laser altimeter reading to the scaled feature range will be the magnitude of translation. This approach requires only one altimeter reading, so it is not susceptible to errors from changing surface relief. Furthermore, it does not depend on nonzero translation along the z-axis. In fact, structure-based scale estimation works better when the spacecraft is descending at an angle with respect to the surface because in this case, scene structure can be estimated more accurately than for pure descent.

The procedure for structure-based scale estimation is to first compute the feature based motion between images along with the depth of the features in the image. Assuming alignment of laser altimeter with the optical axis, the features near the center of the image will be geometrically close to the surface patch that supplies the reading for the laser altimeter (see Figure 3). Since it is unlikely that a feature will correspond exactly to the image center, a few (3-5) features closest to the image center are selected and weighted interpolation is used to determine the scene depth at the image center α_c. The image-based scene depth at the image center has the same depth as the altimeter reading taken when the first image was acquired, so the magnitude of translation is

$$\|T\| = \frac{A_I}{\alpha_c} \tag{11}$$

A number of observations can be made about structure based scale estimation. First, As the translation between images approaches vertical, the structure estimates degrade, especially near the optical axis (i.e., on the optical axis, the displacement between features will be zero for vertical descent - structure from triangulation cannot be computed). Fortunately, vertical descent is precisely the motion where difference scale estimation works best. Second, for the altimeter reading to be related to scene structure, a feature must be located near the optical axis in the first frame, so structure-based scale estimation will work better when more features are tracked.

The magnitude of translation from laser altimetry when combined with feature-based motion completes the 6 DoF motion estimation of the spacecraft.

3 Results on Real Imagery

To test our motion estimation algorithm, we generated two sequences of real imagery. First a comet nucleus analog was created by a comet scientist at JPL. This analog is rough at all scales and matte black, the expected characteristics of comet nuclei. The analog has an approximate diameter of 25 cm. We placed the analog on a rigid stand and took two sequences of images as the camera moved toward the comet analog. The first sequence which

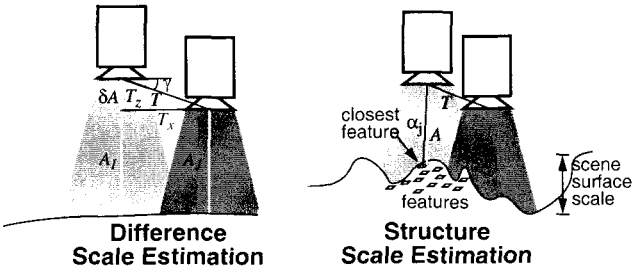

Figure 3: Methods for estimating translation magnitude.

we call *descent* was with a 640x480 CCD imager, a 15 degree field of view lens. The second sequence called *approach* was taken with a 1024x1024 CCD imager and a 25 degree field of view lens. Both sequences were acquired with the camera starting 80 cm from the comet analog; the camera moved 1.00 cm toward the analog between each image.

Ground truth for the image sequence motions were obtained though camera calibration [9]. Each camera was calibrated using a calibration target and as a by product of the calibration procedure, the direction of translation was computed. For the descent sequence, the true translation direction is (0,0,-1), and for the approach sequence, the true translation direction is (0.0096, -0.0033, -0.9999). Since the cameras were rigidly fixed, there was no rotation in the motion.

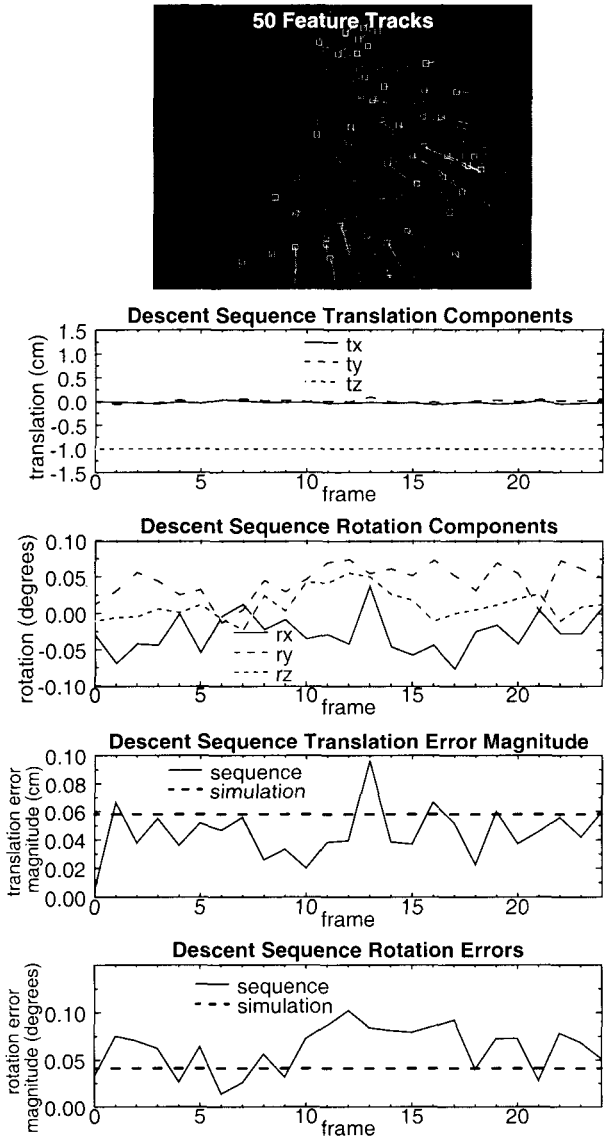

Figure 4: Motion Estimation for the Descent Sequence with 50 features tracked.

An altimeter reading was simulated for each image by using the translation stage reading as the altimeter reading. Using this data type, the scale of translation is know to the accuracy of the translation stage, so no scale estimation method is needed.

The motion estimation results for 50 features and the descent sequence are shown in Figure 4. At the top is shown the feature tracks for the entire sequence. Different shaded tracks correspond to the different key frames when the features were added to the sequence; a key frame occurred every 4 frames. Next are shown the computed translation *(tx,ty,tz)* and rotation angles *(rx,ry,rz)* of the motion computed for each frame using the two stage motion estimation algorithm. Following these is a plot showing the translation error magnitude (vector distance between the true and estimated translations) for each frame in the sequence. On this plot, the dashed line corresponds to the expected performance of the algorithm established using Monte Carlo simulation (assuming perfect feature tracking) for the imaging parameters and motion (See Section 4). Finally, the rotation error magnitude (vector difference between estimated and true rotation angles) is shown for each frame. Again, the dashed line corresponds to the expected performance of the algorithm established using Monte Carlo simulation.

Table 1 summarizes the additional motion estimation results obtained from processing the approach and descent sequences obtained using 50 or 500 features and linear or linear+nonlinear motion estimation

For the 50 feature descent sequence and the linear motion estimation algorithm, the average translation error is 0.045 cm or 4.5% of the distance traveled. The average rotation error is 0.063 degrees from no rotation. These error values are similar to the expected motion errors (0.057 cm and 0.04 degrees) from Monte Carlo simulation given the parameters of the image sequence. The frame rate for this sequence is 4.01 Hz on a 174 Mhz R10000 SGI O^2.

For the 50 feature approach sequence and the linear motion estimation algorithm, the average translation error is 0.028 cm or 2.8% of the distance traveled. The approach sequence results are more accurate because the resolution of the imager is greater The frame rate for this sequence is 2.91 Hz on a 174 Mhz R10000 SGI O^2. The approach sequence takes slightly longer to process because the larger image requires more time to detect features.

The results in Table 1, show that in general the addition of the nonlinear motion estimation algorithm does not improve the results of motion estimation all that much. This is because for vertical descent, the motion computed using the linear algorithm is very constrained, so the results are very close to those obtained using the nonlinear algorithm. Including the nonlinear algorithm in general doubles the running time of the algorithm, so for the vertical descent, it is probably a good idea to remove this stage from the

algorithm if running time is important. However, for other motions (e.g., orbital motion) the nonlinear algorithm will result in improved motion estimation and should be used.

Table 1 also shows that adding features (50 vs. 500) does not improve motion estimation all that much. Since adding features increases the processing time of each frame, suing 50 features is recommended for estimating descent motion.

4 Performance Testing

Using Monte Carlo testing, the effect of sensor parameters (e.g., field of view, resolution), spacecraft trajectory (e.g., motion, altitude) and scene characteristics (e.g., surface scale) on the accuracy of body relative motion estimation can be determined empirically. We used these tests to search for the "best" sensor parameters for precise motion estimation and to predict the performance of the algorithm given a predetermined set of sensor parameters.

4.1 Monte Carlo Simulation

The procedure for a single Monte Carlo trial is as follows: First a synthetic terrain map is generated to represents the surface of the small body. Next, a feature position in the first image is generated by randomly selecting a pixel in the image (feature position in first image). The 3-D position of the feature is found by intersecting its line of sight ray with the synthetic surface. Since the position of the camera for the second view is a known input, the 3-D point can be projected into the second view to determine its pixel position in the second image. Gaussian noise is then added to this feature pixel position to simulate feature tracking errors. This is repeated for however many features are requested. Altimeter readings are computed by intersecting the line of sight for the altimeter (the camera optical axis) with the synthetic terrain, and computing distance between the sensor origin and the surface intersection. Gaussian noise is then added to the range value to simulate measurement noise in the altimeter. Using simulated feature tracks and altimeter readings, the complete 6 DoF motion is estimated.

For these tests some of the motion estimation parameters were fixed: imager resolution was fixed at 1024, field of view was set to at 30 degrees, spacecraft altitude was set to 1000 m, altimeter range accuracy was set to 0.2 m, feature tracking error was set at 0.17 pixels, average feature tracking disparity was set at 20 pixels, scene surface scale was set to 200 m., and number of tracks was set at 500. The remaining parameters to investigate are spacecraft motion and the scale estimation mode used in the algorithm.

4.2 Effect of Motion on Motion Accuracy

This investigation was performed to determine the effect of different spacecraft motions on motion estimation accuracies. To simplify this investigation, the space of possible motions was broken into two groups: descent (pure translational motion) and pointing (pure rotational motion).

Descent can be parameterized by descent angle γ (See Figure 3), the angle between horizontal and the translation direction of the spacecraft. Given the above parameters, simulations showed that a translational motion accuracy of 0.22 m is expected independent of scale estimation mode and descent angle. At a fixed pixel disparity, the distance traveled between frames varies depending on the magnitude of translation. For a horizontal motion (γ=90°), a 20 pixel disparity and 30° field of view corresponds to a motion of 12 m. The motion error is then 0.22 m over 12 m or 1.8%. For a descent angle of γ=45° and a 30° field of view, a 20 pixel disparity corresponds to a motion of 17 m resulting in a motion error of 0.22 m over 17 m or 1.3%. Finally for vertical descent (γ=0°)and a field of view of 30°, a 20 pixel disparity corresponds to a 65 m motion. Thus the error is 0.22 m over 65 m or 0.34%.

By integrating this motion accuracy estimate from multiple frames as the spacecraft descends to the surface an upper bound on the expected horizontal landing position accuracy can be obtained. Simulations showed that the most accurate landing position occurs for the vertical descent with a 10 degree field of view. In this case the landing position accuracy is 3.6 meters. From a height of 1000 meters, this is an accuracy of 0.36% of the starting altitude.

To determine pointing accuracy we only investigated

Table 1: Motion estimation results.

sequence	number of features	motion estimation stages	δT_{seq} (cm)	δR_{seq} (degrees)	processing time (seconds)	number of frames	frame rate (Hz)	δT_{sim} (cm)	δR_{sim} (degrees)
descent	50	linear	0.044927	0.06376	6.24	25	4.01		
descent	50	nonlinear	0.044966	0.0662209	13.1	25	1.90	0.0579763	0.0411912
descent	500	linear	0.033483	0.056666	31.61	25	0.79		
descent	500	nonlinear	0.033615	0.056834	82.33	25	0.30	0.0169	0.0120
approach	50	linear	0.028092	0.024439	2.4	7	2.91		
approach	50	nonlinear	0.023936	0.021443	3.94	7	1.77	0.0659696	0.0505746
approach	500	linear	0.01861	0.017992	13.42	7	0.52		
approach	500	nonlinear	0.018938	0.15937	24.05	7	0.29	0.0221996	0.169442

rotations with axes perpendicular to the camera Z-axis since rotations about the camera Z axis are unnecessary for pointing to surface targets. For a 30° field of view, a 20 pixel average disparity corresponds to a rotation of 0.6° away from the optical axis. Simulations showed that given these parameters, a rotational motion estimation accuracy of 0.006 degrees or 1% of the rotational motion is expected.

4.3 Scale Estimation Mode

Descent angle and scene surface scale dictates which scale estimation mode to use during descent. Simulations were performed to determine at which descent angle the transition between scale estimation modes should occur. This angle is dependent on scene scale and is defined as the angle where translation magnitude errors of the two modes cross over.

The results of the simulation are shown in Figure 5. Inspection of the graph reveals that structure scale estimation should be used except when the surface is very flat (scale < 25 m at 1000 m altitude or 0.25% of altitude) or descent is very close to vertical (γ>88°). Using this plot, it is possible to determine which scale estimation mode to use before scale estimation is performed. Descent angle is fully determined from 5 DoF image-based motion estimation. The scene scale can be determined before descent then though 3-D modeling or analysis of laser altimeter readings. Given this descent angle/scene scale data point, the scale estimation mode can be can be looked up using Figure 5.

5 Conclusion

We have developed and tested a software algorithm that enables onboard autonomous motion estimation near small bodies using descent camera imagery and laser altimetry. Through simulation and testing on real data, we have shown that image-based motion estimation can decrease uncertainty in spacecraft motion to a level that makes landing on small, irregularly shaped, bodies feasible. Possible future work will include qualification of the algorithm as a flight experiment for the ST4/Champollion comet lander mission currently under study at the Jet Propulsion Laboratory. Current research is investigating the use of this algorithm to aid 3-D modeling of small bodies for terrain hazard assessment and comet absolute position estimation.

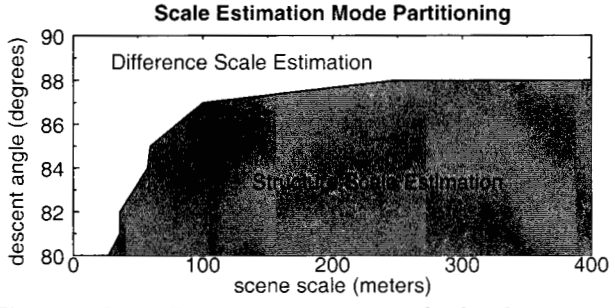

Figure 5: Scale Estimation Mode Partitioning from Monte Carlo Simulation.

The algorithm we have presented can be used to estimate motion with respect to any proximal surface. Consequently, it can be used for precision landing on comet nuclei. asteroids and small moons. It can also be used for proximity operations during rendezvous and docking between two spacecraft. Another application is estimating the attitudinal motion of a orbiter or satellite during precision pointing to surface targets. Rotational motion is completely determined from image-based motion estimation, so a laser altimeter is unnecessary for this application.

References

[1] A. Azarbayejani and A. Pentland. Recursive estimation of motion structure and focal length. *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 17, no. 6, pp. 562-575, June 1995.

[2] A. Benedetti and P. Perona. "Real-time 2-D feature detection on a reconfigurable computer." *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 586-593, 1998.

[3] R. Hartley. "In Defence of the 8-point algorithm." *5th Int'l Conf. on Computer Vision (ICCV'95)*, pp. 1064-1070, 1995.

[4] H. Longuet-Higgins "A computer algorithm for reconstructing a scene from two projections." *Nature*, vol. 293, pp. 133-135, September 1981.

[5] J.K. Miller et al."Navigation analysis for Eros rendezvous and orbital phases." *Journal Astronautical Sciences*, vol. 43, no. 4, pp. 453-476, 1995.

[6] J.E. Reidel et al. "An autonomous optical navigation and control system for interplanetary exploration missions." *2nd IAA Int'l Conf. on Low-Cost Planetary Missions*, Laurel MD, IAA-L-506, 1996.

[7] J. Shi and C. Tomasi. "Good Features to Track." *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 593-600, 1994.

[8] R. Szeliski and S.B. Kang. Recovering 3-D shape and motion from image streams using non-linear least squares. *Journal Visual Communication and Image Representation*, vol. 5, no. 1, pp 10-28, March 1994.

[9] R. Tsai. A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses. *IEEE Journal Robotics and Automation*, Vol. RA-3, No. 4, pp. 323-344, August 1987.

[10] J. Weng, T. Huang and N. Ahuja. "Motion and structure from two perspective views: algorithms, error analysis and error estimation." *IEEE Pattern Analysis and Machine Intelligence*, vol 11, no. 5, pp. 451-476, 1989.

[11] J. Weng, N. Ahuja and T. Huang. "Optimal Motion and Structure Estimation." *IEEE Pattern Analysis and Machine Intelligence*, vol 15, no. 9, pp. 864-884, 1993.

[12] Z. Zhang. "Determining the epipolar geometry and its uncertainty: a review." *Int'l Jour. Computer Vision*. 1997.

We would like to thank Jean-Yves Bouguet for discussions on motion estimation. We would also like to thank Jackie Green for providing the comet analog.